03-20-00 A

jc784 U.S. PTO
03/18/00

WYC:dks

EXPRESS MAIL LABEL NO. EL525676847US
Date of Deposit: March 18, 2000
PATENT
Attorney's Ref. No. 60131-1

jc525 U.S. PTO
09/531076
03/18/00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Box PATENT APPLICATION
TO THE ASSISTANT COMMISSIONER FOR PATENTS
Washington, D.C. 20231

Transmitted herewith for filing is the patent application of:

Inventor(s): Rhoads et al.

For: SYSTEM FOR LINKING FROM OBJECTS TO REMOTE RESOURCES

Enclosed are:

☒ 43 pages of specification, 3 pages of claims,
☒ 10 sheet(s) of drawings.

		FILING FEE				Basic Fee
For	Claims filed	Number Alloted		Number Extra	Rate	$690.00
Total Claims	20	20	=	0	$18.00	0
Independent Claims	3	3	=	0	$78.00	
TOTAL FILING FEE			=			$690.00

☒ Please return the enclosed postcard to confirm that the items listed above have been received.

Address all correspondence to:

William Y. Conwell
Digimarc Corporation
19801 SW 72nd Ave, Suite 250
Tualatin, OR 97062

Respectfully submitted,

DIGIMARC CORPORATION

By _____
William Y. Conwell
Registration No. 31,943

Date: March 18, 2000

Digimarc Corporation
19801 SW 72nd Avenue, Suite 250
Tualatin, OR 97062
Phone: 503-885-8699

SYSTEM FOR LINKING FROM OBJECTS TO REMOTE RESOURCES

Related Application Data

The subject matter of the present application is related to that disclosed in the following applications, each of which is incorporated herein by reference:

09/476,460, filed December 30, 1999

09/343,104, filed June 29, 1999

09/314,648, filed May 19, 1999

60/134,782, filed May 19, 1999

09/292,569, filed April 15, 1999

60/164,619, filed November 10, 1999

60/141,763, filed June 30, 1999

60/082,228, filed April 16, 1998.

Field of the Invention

The present invention relates to data networking, and more particularly relates to systems for linking between objects and associated resources remote resources.

Background and Summary of the Invention

The technology detailed below has many novel aspects and applications. For expository convenience, this disclosure focuses on one particular application – a system for linking print media to electronic content. The technology, however, is not so limited, and may more generally be viewed as a system for linking any object (physical or electronic) to a corresponding networked or local resource.

In accordance with the exemplary application of the present invention, an imperceptible code is embedded within print media, such as magazine advertisements or articles, direct mail coupons or catalogs, bank- or credit-cards, and business cards. When recognized by a suitably-enabled PC camera, that code automatically directs an associated web browser to a destination chosen by the producer of the print media. That destination, e.g., a web page, can provide additional information or services – more

timely and/or more extensive than that provided by the print material. By such arrangement, more efficient internet navigation and access is provided to consumers, and more effective means for linking readers to e-commerce points of sale is provided to advertisers.

The foregoing and additional features and advantages of the inventors' technology will be more readily apparent from the following detailed description, which proceeds with reference to the accompanying drawings.

Brief Description of the Drawings

Fig. 1 is a diagram showing the principal process components of an illustrative system of which the present invention forms a part.

Fig. 2 is a block diagram showing an illustrative system for performing the response process of Fig. 1.

Fig. 3 is a block diagram more particularly detailing an originating device used in the system of Fig. 2.

Fig. 4 illustrates certain top level data flows in the system of Fig. 2.

Fig. 5 illustrates certain data flows associated with the router of Fig. 2.

Fig. 6 illustrates certain data flows associated with the registration process of Fig. 2.

Fig. 7 illustrates certain data flows associated with the product handler of Fig. 2.

Figs. 8-10 show a sequence of screen shots from one application of the present invention.

Detailed Description

Before beginning a detailed exposition, it may be helpful to provide an overview of the larger system of which the present technology forms a part. As shown in Fig. 1, the larger system entails four basic processes – registering, embedding, detection and response.

Registering refers to the process of assigning an ID to an object, and associating that ID with a corresponding action or response. Additional steps can be included, such

as logging the name and/or organization of the registrant, the name of the product, a description of the object and a context in which it is found (magazine, book, audio track, etc.), etc.

Embedding refers to the process of encoding of an object with a digital identifier (e.g., a watermark conveying a serial number in its payload).

Detection is the complementary operation to embedding, i.e., discerning a digital identifier from an object.

Response refers to the action taken based on the discerned identifier.

The middle two steps – embedding and detection – can employ any of myriad well-known technologies, including bar codes, data glyphs, metadata, file header information, RF ID, UV/IR identifiers, organic transistors, and other machine-readable indicia and techniques for associating plural-bit digital data with an electronic or physical object. The detailed embodiment employs watermarking technology, although this is illustrative only.

A great number of particular watermarking techniques are known. The reader is presumed to be familiar with the literature in this field. Particular techniques for embedding and detecting watermarks are detailed in the present assignee's copending application serial number 09/503881, filed February 14, 2000. Other watermarking techniques for images and video are known from published patents to NEC (inventor Cox et al), IBM (inventors Morimoto and Braudaway et al), Dice (inventor Cooperman), Philips (inventors Kalker, Linnartz, Talstra, etc. Audio watermarking techniques are known from published patents to Aris (inventor Winograd, Metois, Wolosewicz, etc.), Solana (inventor Lee, Warren, etc.), Dice, AudioTrack, Philips, etc.

Referring to Fig. 2, a system 10 according to one embodiment of the present invention includes an originating device 12, a router/server 14, a product handler 16, a registration database 17, and one or more remote resources 18.

The originating device 12 can take many different forms, e.g., a cell phone, a personal digital assistant (e.g., a Palm Pilot), a personal computer, a barcode scanning system, etc. For expository convenience, the embodiment is described with reference to a personal computer for device 12.

Device 12 interacts with an object 20. The object can be electronic or not. Electronic objects 20 can include computer files, representations of audio, video, or still imagery (e.g., files or in streaming form), etc. Non-electronic objects can include physical objects such as newspapers, magazine pages, posters, product packaging, event tickets, credit cards, paper currency, etc. Non-electronic objects can also include sounds produced by loudspeakers.

When used with non-electronic objects, device 12 (Fig. 2) typically includes some form of sensor or transducer 22 to produce electronic signals or data corresponding to the object. Examples include CCD- or CMOS-based optical sensors (either as part of still- or video cameras, flatbed scanners, mice, or otherwise), microphones, barcode scanners, RF ID sensors, mag stripe readers, etc. In such cases, the sensor 22 may be coupled to associated interface electronics 24, which in turn may be coupled to device driver software 26, which in turn may be coupled to one or more application programs 28. Device driver software 26 serves as a software interface, communicating at a relatively high level with the application programs 28 (e.g., through API instructions whose content and format are standardized to facilitate application programming), and at a relatively low level with the interface electronics 24.

The detailed embodiment contemplates that the object 20 is a magazine advertisement encoded with a steganographic watermark conveying a plural bit object identifier. The watermark is hidden in the advertisement's image in a manner indiscernable to human observers, but detectable by computer analysis. That analysis is performed by a watermark detector 30.

Watermark detector 30 can be implemented in various different locations in the system of Fig. 1. Typically, the detector is implemented in the originating device 12, e.g., in the driver software 26, or in application software 28c that serves to link to external resources based on detected watermarks. But it may be implemented elsewhere, e.g., in hardware in the interface electronics 24, in an operating system associated with the device, or outside device 12 altogether. Some systems may have plural watermark detectors, implemented at different locations throughout the system.

In an illustrative system, the watermark detector is implemented in the device driver 26. Functionality of the detector is made available to the application program 28c through one or more APIs specific to watermark-related functions. One function is reading of the watermark data payload from the object 20.

The illustrated application 28c is a software program that serves to communicate the watermark data from the device 12 to the router/server 14 through one or more communications links 32 (e.g., the internet). Application 28c also receives information from the communication links 32 and presents same to the user (or otherwise uses same).

The router/server 14 is a high capacity computer including one or more CPUs, memory, disks, and I/O ports. As is familiar to artisans, the disks store operating system software and application programs, together with data, that are transferred to the memory as needed by the CPU. The router essentially serves as a middleman between the application 28c and the product handler 16. As detailed below, the router receives requests from the application, logs them in a transaction log 15, and passes them on to the appropriate product handler.

As more particularly detailed below, the handler 16 provides a response in accordance with the particular watermark payload. The response may be provided directly by the product handler to the device 12, or the handler may respond by communicating with a remote resource 18 (which may be, e.g., a data repository or service provider).

In the former case, the handler 16 may identify a URL corresponding to the watermark (using the database 17), and return the URL to the application 28c. Application 28c can then pass the URL to a web browser 28b in the device 12, and initiate a link to the internet site identified by the URL. Or the handler may have some locally stored data (e.g., audio or video, or software updates) and send it to the device 12 in response to the watermark.

In the latter case, the handler 16 does not respond directly to the device 12. Instead, the handler responds by communicating with a remote resource 18. The communication can be as simple as logging receipt of the watermark message in a remote repository. Or it can be to authenticate device 12 (or a user thereof) to a remote resource

in anticipation of a further transaction (e.g., the communication can form part of an on-line licensing or digital rights management transaction). Or the communication can request the remote resource to provide data or a service back to device 12 or to another destination (e.g., to initiate an FTP file transfer, or to request that a song selection identified by the watermark be downloaded to a user's personal music library, or to update software installed on device 12).

In still other cases, hybrids of the two foregoing cases can be employed, e.g., handler 16 can send some data back to device 12, while also communicating with a remote resource 18.

In some cases, the response returned to the device 12 by handler 16 (or a remote resource 18) can serve to trigger some further action by the device 12. For example, the response returned to device 12 can include a WindowsMedia audio file, together with a request that the device 12 launch the WindowsMedia player installed on the device. (The launching of a browser pointed to a URL is another example of such triggering.)

The illustrated product handler 16 comprises essentially the same hardware elements as the router 14, e.g., CPU, memory, etc. Although Fig. 2 shows just one product handler, several product handlers can be included in the system – either co-located of geographically distributed. Different handlers can be dedicated to different functions (e.g., serving URLs, serving music, etc.) or to different watermark sources (e.g., one responds to watermarks found in audio, another responds to watermarks found in print advertising, etc.). Further specialization may also be desirable (e.g., one handler may respond to advertising placed by Ford, another may respond to advertising placed by Chevrolet; or one handler may respond to advertising appearing in Wired magazine, another may respond to advertising appearing in Time magazine, etc.). In one particular implementation, the router 14 dispatches the incoming data to one of several handlers in accordance with (1) the vendor of the originating application 28c, and (2) the particular identity of the application 28c.

The following discussion focuses on the data exchanged between the application 28c, the router/server 14, the product handler 16, and the associated protocols, in one illustrative embodiment of the invention.

Concept of Operation

When shown a watermarked image, the application 28c analyzes the image and extracts the embedded watermark payload (more particularly detailed below) from the image. The application sends some or all of this information in a message format to the router 14.

The router 14 decodes the received message, looking for vendor and product information. Based on this information, it passes the message to a corresponding product handler 16.

The product handler receives the message and attempts to match the detected watermark serial number to a registered watermark serial number earlier stored in the database 17. If a match is found, the product handler performs the desired action. As noted, typical actions include returning a URL for web redirection, serving up an HTML page for initial user navigation, initiating software downloads, etc. If a match is not found, the product handler returns an error code and message to the application 28c. If a match is found, but the corresponding action is unavailable, incomplete, inactive or invalid, the product handler returns an error code and message to the calling application.

A generalized view of the foregoing is provided in Fig. 4.

(Note that while the system may concentrate on a certain type of object 20, and a certain vendor's application 28c, the architecture is constructed to support accessing product handlers from other vendors and corresponding to other objects. This concept makes the system suitable as a clearinghouse for processing all machine-readable indicia on web-enabled devices.)

An exemplary detection and response cycle is illustrated below.

User	Application	Router	Product Handler
Shows object to sensor 22			
	Acquires watermark Creates message packet Sends packet to product handler		
		Receives message packet Logs transaction	

		Decodes packet Identifies product sending packet Passes packet to Product Handler corresponding to product	
			Logs received packet Validates packet serial number If not found, returns error packet to application Else, returns packet with data/action back to application (e.g., URL)
	Receives packet If error, display error message Else, display data or perform the requested action (e.g., launch browser and link based on received URL)		
Sees the data/action associated with the object (e.g., views web page)			

The present system generalizes this example to support any product from any vendor that is capable of sending a message via the Internet that complies with the expected request format (e.g., a product code, message type, and identifier) and receiving a message in a corresponding response format. One set of message formats suitable for use in such a system are described in more detail below.

Watermark Registration – the first step in the process

In order for the system to identify the response (e.g., a URL) that corresponds to an object identifier (e.g., a watermark), this data must first be associated within the database 17 in association with the watermark to which it corresponds. The watermark registration process captures some basic identification information used later to validate the incoming message, and identifies the associated information/action. In the illustrated example the identification information includes:

- Customer Account,
- Object and associated attributes (name, description, expiration, etc.),

- Action, and
- Registered Serial Number (for registration updates)

The Customer Account identifies the watermark registrant. In most cases, this is also the party to be billed for services. For validation and security reasons, the Customer Account is required to be a known, existing account. Account information, including the account's password, is maintained by an Account Management system.

The Object and associated attributes identifies the object to be watermarked. The object attributes typically include the name and description of the object and a list of accounts authorized to access the object's registration. These authorized "supporting" accounts are typically the ad agencies, pre-press houses, etc. involved in the watermark embedding process in the print advertising example contemplated herein.

The Action defines the response the customer desires when the watermark is detected. It varies by product, but in the illustrative embodiment involves the return of some additional information regarding the watermarked object. In the illustrative system, the action is return of a URL or HTML to be used to display a web page associated with the watermarked object. For other products, the desired response may be display of the object's owner & rights information, software/data downloads, delivery of streamed audio or video, presentation of an advertisement, initiation of object-based actions, etc.

The Registered Serial Number forms the last component of the registration. It is this assigned vendor and product-unique identifier that allows the system to acquire the specific information/action for the object in question.

A few key product Registration concepts -

Watermark Registration is a product-specific process –

To allow each of the products the freedom to upgrade their capabilities without impacting any other product function or schedule, the registration process is product-specific.

Watermark registration is web-enabled -

The exemplary registration is a web-enabled process that requests the basic identifying information from the object owner (publisher, ad agency, studio, etc.) and returns to the registrant a packet with a unique identifier to be embedded within the

object. A watermark embedding application (i.e., software) uses this packet to embed the watermark type and serial number within the client's object. In the illustrative system, only one watermark may be embedded within a single object. In other embodiments, multiple watermarks may be embedded into a single object.

When a customer registers a watermark, the system associates the watermark serial number with the information provided by the customer during the registration process. The associated information may vary with different products. One set of associations, for the exemplary magazine advertisement objects, is shown in the following table:

Mandatory?	Information	Comments
Mandatory	Customer	Typically, the publisher
Mandatory	Publication(s)	Magazine(s) containing the ad
Mandatory	Issue Date	First date of the magazine/publication period
Optional	Volume	Magazine/publication volume information
Optional	Region Code	Optional information for regional publications
Optional	Location Code	Location of the object within the publication (e.g., page and, optionally, finer location data)
Mandatory	Watermark Type	Watermarks may have varying type. The type defines how to interpret the Serial Number
Mandatory	Serial Number	Assigned watermark number
Mandatory	Object Name	Customer's name for the object
Mandatory	Object Description	Customer's textual description of the object
Mandatory	Object Type	Ad or Editorial (and in other systems: Direct Mailer Card, Product Packaging, Coupon, Catalog, Business Card, Credit Card, etc.)
Optional	Campaign	For ads and promotions, the campaign name
Optional	Object size	Stated in fractions of the page (full page, half, etc.)
Mandatory	Effective Date	Date on which the user will first be able to initiate any actions. For publications, typically this is the "on stand" date
Mandatory	Expiration date	Date/time when the watermark expires
Mandatory	Primary Action	Initially the URL used for redirection
Mandatory	Primary Effective	Date the Primary action becomes effective
Mandatory	Primary Expires	Date the Primary action expires
Optional	Default Action	Reserved for future use (e.g., backup to the Primary action)
Optional	Default Effective	Date the Default action becomes effective
Optional	Default Expires	Date the Default action expires
Optional	E-mail address	Used to automatically notify the Customer of problems with the registration/Action
Mandatory	Status	Incomplete, active, inactive
Optional	Problem Indicator	Bad URL, slow site, etc.
Optional	Supporting Accounts	This field and its sub-fields are repeated for each supporting account

Optional	User Fields (4)	
Optional	Text	User Field free text
Optional	Viewable by Others?	Y/N. N hides the field form any other accounts

Table 1. Registration Database Elements

Watermark registrations expire -

For some products, watermarks are granted only for a limited period of time. For these watermarks, the Registration process employs an expiration date for the assigned serial number. When the system receives a message requesting action for a serial number that has expired, an error is returned. Registrants may extend their watermark serial number expirations by updating the expiration date. Expiration extensions may result in customer charges.

Watermark registration can be completed in one or more web sessions -

Registration can be a single or multi-step process. If the media owner has all of the required information at the start of the process, the system can provide a simple web-enabled method for requesting a watermark serial number (s) on-line. With all of the information provided, the registration is considered "active." That is, it is available for immediate use by the consumer. If the registrant does not have all of the required information available at the initial session, by providing a minimum set of information (e.g., name and/or organization name + product), a product watermark serial number may still be issued for the registrant to use in the embedding process. The most typical use of this partial registration occurs when the actions associated with the media to be watermarked (e.g., URL, etc.) are not yet known. The partially registered serial number is considered "inactive" until all of the required registration information has been completed. The system will issue an error message if requested to process an "inactive" serial number. Whether active or inactive, these registrations may be considered billable items subject to the terms and conditions of the applicable contract(s).

Registrations can be updated by the customer to reflect new information and/or to complete a previous registration session. For example, a registered customer may request a watermark serial number without specifying the URL used to redirect the consumer.

The system will assign a serial number so that the customer can continue with the embedding process, but the registration will not be considered complete until the customer updates the registration with the URL and any other mandatory information regarding this serial number.

Watermark registrations are secure -

Only the registrant and those accounts that the registrant authorizes can access specific watermark registrations.

In the illustrated system, the customer account that registers a watermark may grant permission to a specific ad agency and/or pre-press house to change certain fields within the registration as a normal part of their work. Each customer, agency and pre-press house needs an account on file to be granted access to watermark registrations. The Customer account is established as part of the contract process. For ad agencies and pre-press houses, the accounts are established on an as-needed basis, through a controlled web site accessible to the customer.

For all products, the same basic tenet holds – access to the registration information is limited to only explicitly authorized accounts. Accounts are password protected. For ad agencies and pre-press houses, a single password may be shared. In other embodiments, each part may be assigned a unique password.

Watermark registration changes are logged –

All registration actions – creation, modification and deletion - are logged in an audit log. The authenticated username, the date/time of the action, and the action itself are all stored to provide a complete audit trail.

Processes and data flows associated with registration are illustrated in Fig. 6.

Entering Data into the Registration Database

While the client application, router, and product handler have initially been described in connection with responding to watermark information sensed from media objects, the same infrastructure can be employed earlier in the process, to enter data into the registration database 17. That is, a suitably configured variant of application 28c can be used by publishers, ad agencies, pre-press houses, etc., to (a) provide initial data to the

database; (b) update such data; and (c) query the database for current values. Alternatively, a dedicated registration server 19 (Fig. 2) can be employed.

The involvement of plural parties in the registration process can be facilitated by encapsulating the database record contents for a given watermark in a file to which information is successively added (or updated) by different entities, and used to convey data between the database 17 and the cited entities.

Consider a case where Nike advertises in Wired magazine. The ad department at Wired agrees to sell space in response to a request from a media buyer at Nike. Wired may start the related watermark work by securing from the operator of system 10 a particular watermark identifier. (This, and most of the following procedures, are effected by computers talking to computers in accordance with instructions provided by suitable software used by the various participants, etc. In the discussion that follows, this software is the registration server 19 although, as noted, product handler 16 could be arranged to perform these functions.) Wired provides the operator an issue identifier (e.g., San Francisco edition of the July, 2000 issue), and internal tracking information used by the magazine. Registration server 19 responds by sending Wired a confirmatory file, by email, that encapsulates the information thus-far (i.e., the watermark identifier, the issue ID, and the magazine tracking information). Server 19 creates a new database record, and parses the received information into corresponding fields of the record.

Wired forwards the file received from the registration server to the media buyers at Nike. Nike supplements the information with its additional data, including the name of the advertisement and internal tracking information. It then forwards the updated file to server 19. Again, this server processes the file and updates the database record with the new information. It emails a confirmatory data file to both Nike and Wired, so each has the latest set of information.

The process continues in this fashion. Each entity provides new data to the registration server 19 via an emailed encapsulating file. The server updates the corresponding database record, and dispatches updated versions of the encapsulating file to the identified participants so each has the latest information.

Once Nike has entered its data via this process, it may forward the encapsulating file to its outside ad agency. The ad agency uses the file similarly, adding its particular information, and forwarding the file to the server. The server updates the database record accordingly, adds the ad agency to its email distribution list for encapsulating files, and dispatches the latest version of the file to Wired, Nike, and the ad agency.

A pre-press house may be the next party involved, and so forth.

Identification of the URL to which the watermark ID corresponds, and updating of the database record accordingly, may not happen until near the end of the process.

At any time, any of the parties can provide additional information to the database, and share such information with others, via the same process. (Some information may not be suitable for distribution to all involved parties, and can be flagged accordingly.)

Server 19 needn't always be the hub through which all communication takes place. The file as updated by Nike, for example, can be forwarded by Nike directly to its ad agency. The ad agency can add its information, and then provide the twice-updated file to the server, etc.

By using distributed files as proxies for the actual database record, a number of advantages accrue. One is local availability of the latest information by all parties without the need for an internet connection. Thus, if a creative director wants to work on the beach, or otherwise disconnected from the net, the needed information is still available. Another is the ease of integrating software tools at each of the parties with a file of local data specific to a particular advertisement, rather than requiring the architectural hassles of interfacing with a remote database and navigating its attendant authentication and security hurdles.

While the foregoing discussion made reference to emailing files, a typical email program would not normally be used. Instead, to better manage the attendant logistics, a specialized file management/mail program is used by each of the parties. Such program would track the latest file for each advertisement, making same readily available for updating as desired, and index the files by various content fields. The user interface could thus present a list of files, grouped or sorted by any of the database fields, permitting editing or adding of information just by clicking on a given field or tab.

Of course, the file-distribution system just-described isn't essential to the system. A great variety of other arrangements can naturally be employed. One is for each party to log-on to server 19 as needed to inspect, or update, database fields for which it has appropriate permissions.

Numbering Schemes

The payload information encoded into objects (e.g., by watermarking) can take a number of forms and sizes. Four exemplary classes are discussed below:

a) Domain-based payload segmentation;
b) Customer/usage-based payload segmentation;
c) Unsegmented payload; and
d) Unique ID

Domain-Based Payload Segmentation

Domain-based payload segmentation approaches divide the payload into fields, each with a distinct meaning.

Consider a payload of 60 bits. Twelve bits may form a Class ID. These bits serve as an identifier for a top-level domain. 24 other bits may form a DNS ID. These bits identify an intermediate level domain. Together, the Class and DNS IDs fully identify the class of objects from which the data originates, the customer, and the server that should respond to the payload. (Some responses may be handled by the client computer, rather than dispatched to a remote server.)

The remaining 24 bits are a User ID, and serve as the most granular identifier, indicating the particular source of the payload. Based on this ID, the responding server knows exactly which response is to be provided.

This payload is embedded, in its entirety, into the customer's object. When sensed by the client computer, the application 28c parses (decodes) the payload into Class ID, DNS ID and User ID fields. The Class ID is used to trigger one or more of the client- or server-side programs. Once "launched" these products then use the Class ID in conjunction with the DNS ID and the User ID to complete the desired action.

One of the Class IDs may signify the object is a magazine page. Based thereon, the application 28c may direct the payload to the router/handler described above for response. Another Class ID may signify that the object is music. Again, the application may direct the payload to the same router. Or the application may direct the payload to a service maintained by a music industry consortium for response. Still another Class ID may signify that the object is a grocery package, and the payload should be routed to an on-line grocer for response. Yet another Class ID may signify that the object is a business card, and the payload should be processed locally, at the client machine. The mapping between Class IDs, and the corresponding response mechanism to which the application 28c should direct the payload, may be maintained by a database associated with the client computer's operating system (e.g., the Windows Registry), as further detailed in application 09/343,104.

Once the payload has been dispatched to a proper response destination, that entity examines the DNS ID to further classify the correct responding entity. For example, different IDs may correspond to different classes of servers within a tree of servers.

One the payload has been directed to the correct class of servers, the User ID defines the terminal "leaf" in the tree (e.g., a database record) that finally defines the response.

Customer/Usage-Based Payload Segmentation

A second approach again employs a segmented payload technique. In this arrangement, however, a first field defines the interpretation of the following bits (e.g., their segmentation into different fields).

Again, consider an exemplary payload of 60 bits. Twelve bits can be a Version ID. These bits indicate how the succeeding bits are to be parsed and interpreted, and may indicate (like the Class ID in the foregoing approach) the particular application program 28c that should be used. The Version ID bits thus serve to indicate the payload type. In the illustrated embodiment, one of these types signifies that the payload is coming from a magazine page and should be handled accordingly. In this case, the remaining 48 bits

can be parsed into three fields: Owner ID (15 bits), Publication ID (15 bits), and Media ID (18 bits).

The Owner ID identifies the customer to whom the watermark is registered (e.g., Nike). This is used for ad effectiveness analysis and billing purposes. The Publication ID identifies the particular publication (e.g., July, 2000, San Francisco edition of Wired Magazine). The Media ID identifies a particular page location within that publication.

As before, the payload is embedded in its entirety into the customer's object. The payload is first parsed to determine the Version ID. If the user's device 12 has been programmed to handle such objects locally, further parsing is performed in accordance with data corresponding to that Version ID, and associated processing of the parsed data is performed. If the device has been instructed to dispatch such payloads to remote locations for service, the complete payload can be dispatched with only such further parsing (if any) as may be required to correctly identify the corresponding remote servicing entity.

Unsegemented Payload

An unsegmented payload consists only of two parts: a Version ID (as described above) and an Object ID. In an illustrative case, a 60-bit payload is again used, with 12 bits serving as the Version ID, and the remaining 48 serving as the Object ID.

In this approach the relationships of owner/customer, publication, issue, and media are all maintained in database 17 rather than literally represented in some fashion within the object identifier.

Unique ID

This case is akin to the unsegmented payload, but consists of just a single field – a unique identifier. The same application 28c is always used, and always treats the payload data consistently (e.g., processing locally, or dispatching to a predetermined destination) regardless of the payload contents.

Combinations and hybrids of the foregoing approaches can of course be used. Moreover, the 60 bit payload length is illustrative only. Longer (e.g., up to 1024 bits) or shorter (e.g., down to 8 bit) payloads can naturally be used.

In a particular embodiment, a 31-bit unsegmented payload is used, consisting of 9 bits of payload type and 22 bits of watermark serial number. Some materials (e.g., advertisements including composite graphics) may be encoded with several serial numbers. The mapping between this payload and the customer/publication/etc., is maintained in database 17.

(As noted below, the data sent from the application 28c typically includes information other than the identifier payload, e.g., the type and version number of the application 28c, the electronic address of the dispatching application, etc.)

Router

The router 14 permits any number of different products to be used by the indicia detection and response model. By keeping this function separate and generalized, new products can be added without design changes to the existing products or the product handlers 16. There are two keys to making this approach successful – speed and flexibility. By using a standardized, open interface, the router is able to facilitate both of these goals.

A premise of an exemplary interface is an enveloping technique that allows the router to "open" the outer transaction envelope and extract the vendor and application ID without decoding the remainder of the transaction (message). Given these two pieces of information, the router uses a simple lookup table to determine the product handler appropriate to complete the transaction. The router then passes the vendor, application, remainder of the transaction and the Internet "reply to" address on to the appropriate product handler. The simplicity of this handling keeps the routing delay to a minimum, while deferring the actual response processing to vendor/product-specific handlers. By including the "reply to" address in the data passed on to the product handlers, the router is freed from the responsibility of return routing for the product's response(s).

To review, the router:

1. decodes the request packet received from the client products into the packet's base components – Vendor ID, Application ID and message;

2. validates the request packet base components against a list of known, good values;

3. if a request packet component is found to be invalid, issues an error message noting the invalid components and returning same to the calling session (e.g., product);

4. sends the decoded request packet contents and any required identification of the calling session to the appropriate product handler; and

5. reports any errors encountered, including invalid packets received, to a system monitor.

Certain data flows associated with the router are shown in Fig. 5.

Product Handler

The primary function of the illustrative product handler 16 is to process requests received from the application 28c, via the Internet and router 14, and return the requested information/action to the originating device 12. In the illustrated embodiment, the information requested is the URL associated with the watermark payload sent by the application. In other embodiments, other actions and/or information may be requested.

Each received watermark payload is validated using information in the database 17. If the watermark payload ID is found and is active, the requested action is performed. If the watermark payload ID is not found or is in an inactive state, an error message is returned to the requesting application.

All requests are logged in a transaction log for tracking and billing purposes. This includes any secondary payload information (zip code, Demographic Household ID, etc.) passed in by the application 28c. The log can be maintained by the product handler 16, or elsewhere.

To speed system response, the product handler 16 may anticipatorily send URLs to the application corresponding to watermark payloads the handler foresees may be coming. These URLs can be cached in memory associated with the application 28c, and quickly recalled if needed by the application.

Consider, for example, a magazine containing watermarked advertising. If the user presents a first ad to the device 12, the watermark is decoded and forwarded to the product handler 16, which responds with a URL corresponding to that ad. The application 28c then passes that received URL to a web browser 28b on the device 12, which initiates a link to that internet address. But the handler now knows the magazine the user is reading. By reference to the watermark first received, the handler may discern, for example, that the user is reading the San Francisco edition of the March 14, 2000, Time magazine, and just looked at page 85. Based on this information the handler can query the database 17 for URLs associated with other advertising in that issue. (The database index is structured to permit fast queries identifying all ads in a given magazine issue or other collective data source.) These URLs are passed back to the application 28c and cached. If the user next presents an advertisement from page 110 to device 12, the application 28c finds it already has the corresponding URL locally cached. The application then passes the corresponding URL to the web browser. The web browser initiates the link immediately, obviating a data round trip between the application and the remote system.

The caching can be optimized in a variety of ways. One is to first send URLs corresponding to pages that are next-expected to be encountered. For example, if the user just presented page 85 to the sensor 22, after sending the URL for that page, the handler 16 would next send the URLs associated with pages 86, 87, etc. On sending the URL for the last page of the magazine (typically the rear cover), the handler could start from the beginning (typically the front cover) and send further URLs up to that for page 84. Another optimization is to first cache URLs for the most conspicuous ads, e.g., first send URLs for any 2-page spread ads, then for each full page add, then for each successively smaller fractional-page ad. Still another approach is for handler 16 to dispatch URLs to device 12 for caching in accordance with a contractually-agreed priority. One advertiser, for example, may pay a premium ad rate in exchanged for being cached before other advertisers who don't pay the premium. Other caching priorities, and combinations of such priorities, can naturally be employed.

In some systems, the advertisers or publishers may be charged for use of the system based on the number of URLs served by the system for linking. If local caching of URLs (e.g., at device 12) is employed, it is desirable for device 12 to report to router 14 (or handler 16) the URLs that are actually retrieved from the local cache and used for linking, so that the remote system can log same. Thus, each time the user presents an object to sensor 22 for which a corresponding URL is already cached, application 28c dispatches a message to router 14 reporting the event (and, usually, the particular URL involved). This event is then logged in the transaction log.

This anticipatory dispatching of URLs is one alternative function that may be performed by a product handler. Another is if the application 28c queries the product handler to determine if a more recent version of the application is available for download. If so, the application – through interaction with the user – can request that the product handler respond with a software download.

In greater detail, application 28c can periodically query the product handler as to the identity of the latest version of application 28c (e.g., the first time the application is used each day). Device 12 may have version 3.04, and the remote system may respond that version 3.07 is the most current. In such case the application 28c can alert the user – by suitable text, graphics, or other means – that a more recent version of the program is available, and query whether such updated version should be obtained. If the user so-instructs, handler 16 can serve to device 12 the latest version of the application (or a patch permitting the presently-installed version to be updated).

Sometimes it may not be necessary to update the application version. Instead, data from the remote system may indicate the desirability, or necessity, or changing just one or more parameters in the application 28c. For example, new security keys can be dispatched periodically by handler 16 to device 12, and used to change the security configuration of the application. Or the application 28c can be instructed to direct further outgoing watermark traffic – either for the next hour, day, or until instructed otherwise - to a different router 14. Such instruction can be used to optimize system performance, e.g., for router load balancing purposes, to avoid internet routes that are found to be slow, etc.

In summary, the detailed handler:

1. validates the received identifier (e.g., watermark serial number) against the list of active identifiers; and, if the serial number is not found, return an error message to the calling session, and log the error to an error handling routine;

2. for each received, valid watermark serial number, finds the corresponding active primary action from the database;

3. for each received, valid watermark serial number, if the handler finds the corresponding primary action is currently not active, it performs an alternative, "default" action instead;

4. if the handler finds an active primary action associated with the received valid watermark serial number, it returns the URL for application use in redirection (round trip approach), or serves the HTML page found to the calling session;

5. if the handler does not find an active primary action associated with the received valid watermark serial number, but does find an associated default action, it returns that URL for application use in redirection (round trip approach), or serves the HTML page found to the calling session;

6. if the handler does not find a valid, active primary or default action associated with the watermark serial number, it returns an error message to the calling session, and logs the error to the error handling routine;

7. records each transaction, including those that result in error messages, for billing and analysis purposes (in other embodiments, this function may be performed by the router, instead);

8. responds to a "software version request" by returning the most recent available application software version number to the calling session;

9. responds to a "software download request" by initiating a file transfer of the most recent available application software to the calling session;

10. responds to a valid Request for Registration packet upload (proper format, an existing serial number, an account ID and a valid corresponding account password) by returning a current registration packet for the provided watermark serial number;

11. responds to an invalid Request for Registration packet by returning an error message to the calling session noting the failure;

12. responds to a local transaction cache flush request by writing the locally cached transactions to the transaction log; and

13. responds to a multiple URL request by returning the URL associated with the provided serial number first, followed by all other active serial numbers and URLs for the publication, issue and region code (optional) provided.

Certain of the above-described processes associated with the product handler are shown in Fig. 6.

URL Performance Monitoring

Returning to operation of the system, the URLs identified in database 17 may, from time to time, become inoperative or impaired due to equipment problems at the remote web site or otherwise. If desired, the handler 16 (or another component of the system) can be programmed to periodically test each of the links registered as active in the database (e.g., once per day), and measure the time for the associated web page to load. If the web page doesn't load, or takes much longer to load than usual (and re-tests confirm that the condition isn't an anomaly), those conditions can be flagged in the corresponding database record. If the handler is requested to provide such a URL to a device 12, the handler can send a message – either with or without the URL – indicating to the device that the URL is misbehaving.

If the URL is working, but is unduly slow to load (either compared to its historical performance, or compared to other URLs), handler 16 can provide an interim diversion to the device 12. For example, it can instruct the device to launch a second browser window, and direct that browser to an alternate destination to entertain the user while waiting for the intended page to load. When the intended page is finally loaded, the first browser window can be displayed - either by closing the second, diversionary window, or by bringing the first window to the front while keeping the second window alive in the background.

This alternate destination is desirably a low bandwidth page, so that it will not unacceptably further slow loading of the desired URL. This alternate page can be one selected by the handler, for which the URL is sent after the desired URL. Or instead of providing a URL from the handler, the handler can serve an HTML or other page directly to the device 12. Or the alternative URL can be stored at device 12 and used to invoke the second browser window upon receipt of data from handler 16 indicating that the desired content will be slow in coming. In some embodiments the user can identify several alternate URLs (e.g., weather, stock info, jokes) and the handler or the application 28c may select among them randomly or otherwise. Or an HTML page or other application can be loaded locally at the device 12 in response to a "get ready to wait" indication from the handler 16.

If a URL is marked in the database 17 as slow or inoperative, the scanning operation periodically rechecks the URL to see if its status in the database should be changed (e.g., changed from inactive to active). Inactive URLs are reported to the registrant by email, and flagged for manual follow-up if not restored to action within a predetermined period.

Illustrative Responses by Product Handler

The reader is referred to application 09/343,104 for a sampling of the great variety of diverse applications enabled by the illustrated system 10. A few more are detailed below.

Consider use of the system 10 to enable personalized greeting cards. A greeting card company may prepare watermarked press-on stickers for use with its cards or other correspondence. The customer shows the sticker to a camera-equipped computer (either at the retail store, at home, or elsewhere). The computer decodes the watermark and sends same to a corresponding product handler 16 through the router 14. The handler – recognizing the watermark as an unregistered greeting card sticker – invites the customer to enter a destination URL, such as the customer's personal web page. This information is entered by the consumer and forwarded to the remote system for entry in the registration database 17. Thereafter, whenever the sticker is shown to a suitably-enabled

system (e.g., by the card recipient), a browser window is automatically launched and directed to the web page specified by the purchasing consumer. (The same result can, of course, be effected without use of stickers, e.g., by encoding the greeting cards themselves.)

In some applications, the product handler may have a library of different responses it can provide to a user in a particular context, depending on the user's further selection. Consider a university student having a suitably-watermarked university ID card. When the card is presented to a device 12, the product handler replies with HTML instructions causing an options menu to appear on the device screen, e.g:

1. Review calendar of upcoming university academic events
2. Review calendar of upcoming university sporting events
3. Review present class schedule
4. Select courses for next semester
5. Review grades

When the student makes a selection (e.g., with a mouse, or by moving the ID card in a specified manner), the application 28c dispatches data corresponding to the selected option to the product handler, which then responds with the requested data.

In some cases (e.g., Review present class schedule, Select courses for next semester, Review grades), care must be taken to protect such information from persons attempting access using lost or stolen IDs. Accordingly, when any of these options is selected, the handler 16 may first respond to device 12 by querying for a password or PIN. Only after entry of the correct password/PIN is the requested action performed. (For security reasons, the university may prefer that the password authentication process be performed by a dedicated on-campus server, rather than by product handler 16. Naturally, this and other tasks can be delegated to processors other than handler 16 as best fits the situation.)

In other cases, an option menu needn't be presented – the correct response is inferred from the context or environment. Consider a drivers' license that is watermarked with identification of the owner. If presented to an email kiosk 12 at an airport, the decoded watermark may be used to look-up an email account corresponding to that

individual, and download new mail. If the same drivers license is presented to a check-in kiosk, the decoded watermark may be used to look up that person's flight reservation and issue a seat assignment. In both cases the kiosks can be essentially identical. One, however, identifies itself to the router/product handler as an email kiosk, the other as a check-in kiosk. The response undertaken by the router/product handler differs accordingly.

Returning to the university example, there may be cases in which students are tempted to swap photos on a student ID, e.g., to permit an imposter to take a graduate school qualifying exam on behalf of a less-qualified student. In the usual case, such photo-swapping may be difficult to detect. This problem can be combated by an exam check-in procedure that includes having each student present their ID to a device 12. An application 28c specialized for this task can forward a watermark decoded from the ID photograph to a handler 16, which responds by causing an image of the identified student to be displayed on device 12. (The university could compile the requisite database of student images as it issues ID cards.) If the exam proctor sees an image on the device that does not match the image on the ID card, appropriate action may be taken. (This arrangement is applicable wherever photo ID documents are used, including airport check-in, customs, immigration, etc.)

Still another application of the illustrated system is to look-up, or act on, meta-data associated with a marked object. Consider an image, video, or audio file that a user downloads from the internet. Familiar applications such as Microsoft's Windows Explorer (including Internet Explorer) may be configured with watermark decoders activated, e.g., from a Properties panel (accessed, e.g., by right-clicking on the file icon or name and selecting the "Properties" option). When a watermark is detected in a file, the Explorer application can send a corresponding packet to a remote system (e.g., the depicted router/product handler/database). The remote system recognizes the packet as originating through the Properties panel of Windows Explorer, and looks-up the watermark ID in a database 17. Meta-data corresponding to the file (e.g., proprietor, creation date, licensing terms, exposure data, subject, etc.) is returned from database 17

(or from another database identified by the router, handler, or database) to the application 28c, and is displayed in the Properties panel (optionally under an appropriate "tab").

(The present assignee has long offered a "MarcCentre" service that serves as a clearinghouse through which watermark identifiers found in photographs, etc., can be used to identify the proprietors and associated information corresponding to such objects.) In embodiments of the present utilizing this service, the router 14 passes the request to MarcCentre server (a product handler in this instance), which provides the solicited information back to the originating application. The present assignee's MarcSpider service complements the service provided by the Media Commerce product. The MarcSpider service constantly scans Internet sites evaluating each graphic encountered to determine whether it contains a watermark. (Audio and video can be similarly analyzed.) With each detected watermark, the MarcSpider service records the graphic file name, size, format, date/time and URL where the graphic was found. This information is then made available to MarcSpider customers in report form.)

Instead of simply displaying the meta-data, the application and/or the remote system can make use of it. For example, if the meta-data indicates that the proprietor of a watermarked image is Corbis, and that the image can be licensed for a certain use under certain terms, the remote system can be utilized as a licensing server – receiving payment information from the user, granting the license, and forwarding transaction details to Corbis.

Still another application is the sale or promotion of music or video over the internet. Taking the case of music, an artist may freely distribute a low-fidelity (or otherwise corrupted or abridged) version of a song. The low fidelity can be by reason of bandwidth limitation (e.g., 500Hz – 2.5 KHz), monophonic (as opposed to stereo), or otherwise. The artist can seek to distribute the low-fidelity version as widely as possible, to serve as a marketing agent for the artist's other works. (The free distribution of lower-bandwidth audio may serve to alleviate some of the network bandwidth problems faced by universities whose students actively engage in transferring free music over the internet.)

Each low-fidelity version can be processed to extract an identifier (e.g., a steganographic in-band watermark; a numeric ID or song/artist name field in a in a file header; a 128-bit hash value obtained by applying a hashing algorithm to the music data, the music file header data, a portion thereof, etc.) If a listener is interested in obtaining a full-fidelity version of the work, the listener can operate a suitably programmed computer or music appliance that extracts the identifier from the work and passes it on to the remote system. The remote system can respond in various ways, e.g., by providing a full-fidelity version of the same work back to the user (such as MP3 download) and charge the user's credit card a fee (e.g., $0.99); or by directing a web browser on the user's computer to an e-commerce/fan web site associated with the music, etc. Such functionality can be provided in general purpose programs such as Microsoft's Internet Explorer, e.g., by right-clicking on a file to obtain a menu that includes this and related functions.

Figs. 8-10 show a sequence of screen shots from such an embodiment. In Fig. 8, a user has right-clicked on an MP3 file icon in a directory listing 200. A property menu 202 pops up that includes, as its second option "MP3Bridge."

Fig. 9 shows what happens when the user selects the MP3Bridge option. An MP3 player 204 is launched, and a dialog box 206 appears. The dialog box queries the user, "More Information About the Artist? Yes No."

Fig. 10 shows what happens if the user selects "Yes." The software sends the identifier – extracted from the MP3 file – to the remote system. The remote system responds with the address of an associated web page, and instructs the user's computer to launch a new browser window directed to that page.

The same functionality can naturally be invoked through the user interface of the MP3 player (or a Windows MediaPlayer, etc., rather than through Internet Explorer). The music application can spawn a separate window, or present the options and the associated data within the existing window.

Yet another application of the remote system is as a "net nanny" filter. Links requested through the system can be checked for keywords, adult-content flags, content

ratings, or other indicia of age-suitability, and provided to the requesting computer 10 only if they meet certain earlier selected criteria.

Again, it will be appreciated that the foregoing examples are but a few of myriad applications enabled by the detailed system.

Reporting

System software may enable the provision of customer-accessible reports (accessible over the internet) that show detailed and summary usage information by date, customer, publication, issue date, region, product/version, etc. These reports can be both regularly scheduled and ad-hoc. The specification of the content, relationships and the timing of the reports can be defined by the customer on-line.

Illustrative reports detail:

a) Hit rates/Transactions per customer per ad
b) Hit rates/Transactions per customer per publication per ad
c) Hit rates/Transactions per customer per publication per issue per ad
d) Hit rates/Transactions per customer per publication per issue per region per ad
e) Hit rates/Transactions rates by originating application (28c)
f) Hit rates/Transactions by originating application vendor
g) Hit rates/Transactions rates by originating web domain (e.g., aol.com)
h) Hit rates/Transactions rates by postal/zip codes
i) Hit rates/Transactions by country

Additional marketing/marketplace reporting can also be produced for internal analysis by the service provider, and for sale to other entities. These reports typically provide a more global view of the impact and usage of the system. Using information stored in a demographic database, in conjunction with these usage patterns, the system can provide customers and research agencies with more detailed demographic/statistical data on the system's usage and effectiveness.

In an illustrative system, certain statistics in the demographic database are compiled using statistics from a sample of users that consent to have their activities

tracked in some detail, in consideration for certain perks (e.g., give-away cameras, bar-code scanning pens, or other devices, etc.). These users are termed Demographic Households. A software program included in the systems solicits information detailed in the following table from such users over the internet, with a web-enabled interface. A related program allows such users to update/edit their user/household information previously entered. Each such session is password authenticated for security.

User Information	Comments
Name	
Address	
Street	
City	
State	
Country	
Postal Code	
Phone number	
E-mail address	
Household Annual Income	Provided as raw $ or as a selection from a range of numbers
Occupation	
Education	May be per member of household.
Profession	If applicable
Number of members of household	
Member of household	
Age	
Sex	
Internet user?	
User of this linking service?	
Internet usage per week	In hours. Sum of entire household
Internet business usage per week	
Primary Internet usage?	Typical household use of the Internet. May be a selection list
Owned a computer since?	Year only
Number of computers in the home?	
Types of computers in the home?	Mac, PC, etc. Select all that apply
Rooms where the computers are located	Home office, bedroom, etc. Select all that apply
What ISP do you use?	
What is your modem speed?	Select from list that includes ISDN, ADSL, cable + dial up modems.
Are you willing to be an official "Demographic Household" and allow us to contact you for feedback on our products and advice on new products?	
What other technology devices do you have in your household?	Scanners, PC cameras, digital cameras, DVD, PDAs, etc. Select all that apply

Audio and Video

As with paper advertisements, the illustrated system provides users of web-connected PCs or other appliances with methods of obtaining information, content, associated products, or associated services using the same principles as detailed above.

For example, an application 28 can "capture" music or other audio using a recording device (note recorder, microphone connected to PC, MP3 player, etc.) and analyze the captured audio to detect an embedded watermark. Once detected, the application passes some or all of the watermark payload information, together with identification of the application and its vendor, to the router. The router forwards the payload information to a handler corresponding to the application. The response of the product handler varies with the context and nature of the data. For example, the handler may return the artist, title, track, album, web URL and purchasing information, to the user. Recorded news and entertainment segments may include transcripts (audio, video and/or text) of the segment along with other related web site information. The handler may simply cause the device 12 to launch a browser window directed to a music commerce web site where the music can be purchased.

Audio and video applications of the present technology are more particularly detailed in the related applications cited above.

Monitoring

Automated performance monitoring occurs at several levels within the detailed system, including:

a. Network monitoring (traffic + device availability)
b. Computer system monitoring (disk space, memory utilization, CPU usage, etc.)
c. Process monitoring (scheduled job starts, stops, run times (duration), and errors)

For each monitored point, two thresholds are typically established – warning thresholds and failure thresholds.

When a system exceeds a warning threshold value, it indicates that a situation is now present that may affect the system's performance in the near future. These warnings are used by the operations staff to help head off potentially serious problems. While the warning indicators are available at all times for operations staff viewing, they do not require immediate attention.

When a system exceeds a failure threshold, it means that a condition that directly affects the system's ability to properly serve its customers has been detected. A monitoring system immediately notifies the appropriate personnel, via pager, of the condition. If the condition is not resolved or acknowledged within the pre-defined elapsed time, the monitoring system escalates the problem notification to the backup personnel and to company management.

All monitored activities are logged (successful completion, errors, warnings and failures) for ongoing system reliability analyses.

Backup and Recovery

While the illustrated architecture desirably employs redundant systems to increase the overall reliability and availability, rapid recovery from failures is required for peak system performance. Areas of possible system failure include

a) Loss of computer
b) Loss of network/network segment
c) Loss of electrical power
d) Loss of Internet connectivity
e) Loss of physical site
f) Corrupted software
g) Loss of data/database
h) Corrupted database

The system has specific backup and recovery plans and procedures for each of these conditions:

a. To address the loss of computers, the architecture is built with redundant capabilities.

b. To address the loss of network/network segment, the computers are simultaneously connected to two separate network segments.

c. To address the loss of electrical power, the physical facility is provided with plural electrical feeds from different power substations entering the facility from plural directions.

d. To address the loss/corruption of software and data/databases, the critical software and data are housed on redundant computers and backups of these are regularly scheduled.

e. To address the loss of Internet connectivity, the facility has plural separate Internet connections provided on two different telecommunications trunks routed into the building from different directions.

f. Finally, to address the loss of physical site, the system is desirably mirrored at a remote location, with the capability to direct all traffic to the mirror site until operations at the primary site are restored.

Security

The system's basic security philosophy is to grant access to each customer's information only to the users authorized by the customer. To this end, the system desirably should:

1. Create and maintain a list of authorized users (accounts).
2. Employ security methods to deny access to any unauthorized users.
3. Limit users to access only the objects they are authorized to access (typically, the objects belonging to that customer).
4. Report and record all unauthorized access attempts.
5. Maintain a log of all authorized user logins (sessions).
6. Provide the capability for the watermark registrant to grant access rights to other accounts (such as ad agencies and pre-press houses).

7. Establish initial passwords for each account

8. Provide the capability for each authenticated user/account to change their password

9. Provide the capability to reset an authenticated user/account's password in the event the current password is lost.

10. Store all passwords as encrypted values (to prevent theft of passwords).

11. Provide the capability to restrict the creation, modification, deletion, and listing/viewing of account information to authorized users.

Audit Trail

Because of the financial implications of the system's activities, all changes to any registration or customer data need to be recorded. This audit trail provides the operator and its customers with an accurate accounting for the current and previous states of the data.

The audit software desirably records the creation, modification, and deletion of all registration and customer data. The audit software also records the username, date/time of creation/modification/deletion of records, and – for modifications – the before and after images of the data changed.

Application-to-Product Handler Interface Definition

The basics of the interface between the application 28c and the handler 16 are (a) a flexible request and response package structure, and (b) a defined connection method based on industry standards. The illustrated messaging employs the http and/or the https protocol to send and receive messages among the system components. An overview is provided in Fig. 4.

Message Format

The message format is XML-compliant and is defined by the following XML DTD –

```
<!DOCTYPE list [
<!ELEMENT Content (vendor, appl, prod)>
<!ELEMENT vendor (#PCDATA)>
<!ELEMENT appl (#PCDATA)>
<!ELEMENT prod (#PCDATA)>
]>
```

The application 28c appends its data to this header for transmission to the product handler 16. Exemplary messages and product handler responses are detailed in the sections that follow.

Application Message Definitions

The application message definitions can be broken down into Request Code, Primary and Secondary information.

The **Request Code** instructs the product handler 16 to take a specified action.

The **Primary information** portion contains the data required to properly service the application's request. The Primary Information varies based on the Request Code.

The **Secondary Information** is intended for use by analysis and reporting tools and does not instruct nor aid the product handler in servicing the user's request. Secondary Information contents change based on the Request Code and not all Request Codes are required to have associated Secondary Information. In addition, most of the Secondary Information requires the consumer to grant express consent to its collection. If that consent is not given, the application does not send Secondary Information. A special case exists for selected, consenting consumers to become part of a demographic database.

Primary and Secondary information may change by request type, but in general conform to the definitions below. The generic format for the product handler is also defined below.

Primary Information includes the Application Version, Watermark Type, Watermark Serial Number, Context and Environment.

- Application Version: used by the product handler to modify its actions, typically for backwards compatibility

- Watermark Type: top 9 bits of the illustrative watermark payload. Used by the product handler in processing the Watermark Serial Number
- Watermark Serial Number: remainder of the watermark payload. Provides the index used by the product handler to access the watermark in the registration database
- Context: instructs the product handler to modify/refine its action based on the consumer request's context
- Environment: instructs the product handler to modify/refine its action based on the consumer request's environment. (The environment may be specified, e.g., as home, office, car, portable appliance, etc.)

Other Request codes can, of course, be used. Each may have its own list of mandatory and optional Primary Information fields. Optional fields are excluded from the primary Information when there is no value associated.

Secondary Information:

- Demographic Household ID: identifier for a selected demographic group. This is used as an index to the actual demographic
- Input device: Manufacturer, model and version of the device used to detect the watermark (e.g., a TWAIN driver string)
- Operating System: operating system in use on the consumer PC
- Processor: processor type/class on the consumer PC
- Processor speed: processor clock speed, in MHz, of the consumer PC. (May be entered by user, or auto-detected.)
- Language: preferred consumer spoken language
- Country: Country where the consumer PC resides
- Postal Code: Consumer's postal code (used along with the country to pinpoint the location of the consumer).

(In addition to these explicit data, the packet sent from the device 12 also conveys an IP address (inherent in the use of http protocols) so that the remote device (e.g., the router/handler) has an address to which it can respond.)

Response from Product Handler

```
RtnCode          - Success =1
URL              - the active URL for the watermark serial
                 number received
                       or
RtnCode          - Error <0
Error Message    - text.
```

Request for URL

Required Inputs

Header (XML format)

```
Vendor      (e.g., = Digimarc)
Appl        (e.g., = MB)
```

Data

Required information –

```
Req         =RFU
Ver         = application version number
Type        = watermark type number
Ser         = watermark serial number
Cxt         = context
Env         = environment
```

Optional Information –

```
Ctry        =User's Country name
Lang        =User's preferred Language
HHID        =Demographic Household Identifier
Det         = TWAIN string of the sensing/detecting device
OS          =User PC Operating System string
Proc        =User PC processor type and class
Speed       =User processor speed
Zip         =User postal code
```

Example:

```
<?xml version="1.0"?>
<Content>
<vendor>Digimarc</vendor>
<appl>MB</appl>
</Content>
Req=RFU
```

```
Type=1
Ser=10001
Ver=1.0
Cxt=A
Env=Q
Ctry=USA
Lang=English
HHID=1234567
Det=TWAIN string
OS=Win98
Proc=Pentium III
Speed=500
zip=74008-1234
```

Response from Product Handler

```
RtnCode=Success/Error number (Success = 1)
URL=URL associated with specified watermark type and Serial number
Exp=Expiration date/time (GMT) for caching purposes - format of
mm/dd/yyyy hh:mm:ss
      or
RtnCode=Success/Error number (Error <0)
MsgText=message text
```

Error reasons:

-1 Type and Serial Number OK, but no URL in database (both the primary and default URL are missing)

-2 Type and Serial Number OK, but URL is marked as inactive (neither the primary nor the default is active)

-3 No record in database matching the Type and Serial Number

-4 Request format error – incomplete data

Request for Configuration

Required Inputs

Header (XML format)

```
Vendor      (e.g., = Digimarc)
Appl        (e.g., = MB)
```

Data

Required information –

```
Req         =RFC
OS          =User PC Operating System
```

Example:

```
<?xml version="1.0"?>
<Content>
<vendor>Digimarc</vendor>
<appl>MB</appl>
</Content>
Req=RFC
OS=Win98
```

Response from product handler

```
RtnCode= Success/Error number (Success = 1)
Ver=Latest Application version# available for download
https=yes (or n )
GCURL=URL used to route subsequent Application requests
      or
RtnCode= Success/Error number (Error <0)
MsgText=message text
```

Error reasons:

-5 Unknown Operating System

-4 Request format error – incomplete data

Request for Associated URLs

Required Inputs

Header (XML format)

```
Vendor      = Digimarc
Appl        = MB
```

Data

Required information –

```
Req         =RFA
Ver         =application version number
Type        =watermark type number
Ser         =watermark serial number
Cxt         =context
Env         =environment
```

Example:

```
<?xml version="1.0"?>
<Content>
<vendor>Digimarc</vendor>
<appl>MB</appl>
```

```
</Content>
Req=RFA
Type=1
Ser=10001
Ver=1.0
```

Response from product handler

```
RtnCode= Success/Error number (Success = 1)
Ser1=watermark serial number
Type1=watermark type number
URL1= URL associated with specified watermark type and Serial
number
Exp1=Expiration date/time (GMT)
Ser2=watermark serial number
Type2=watermark type number
URL2= URL associated with specified watermark type and Serial
number
Exp2=Expiration date/time (GMT)
…..
Ser'n'=watermark serial number
Type'n'=watermark type number
URL'n'= URL associated with specified watermark type and Serial
number
Exp'n'=Expiration date/time (GMT)
      or
RtnCode=Success/Error number (Error <0)
MsgText=message text
```

Error reasons:

-8 Type and Serial Number OK, but no associated watermarks or URLs in database

-9 Type and Serial Number OK, but all associated URLs are marked as inactive

-3 No record in database matching the Type and Serial Number

-4 Request format error – incomplete data

Request for Transaction Download

(Needed to account for locally cached redirections. One request per local redirection.)

Required Inputs

Header (XML format)

```
Vendor      = Digimarc
Appl        = MB
```

Data

Required information –

```
Req         =RFT
Ver         =application version number
Type        =watermark type number
Ser         =watermark serial number
Cxt         =context
Env         =environment
```

Optional Information

```
Ctry        =User's Country name
Lang        =User's preferred Language
HHID        =Demographic Household Identifier
Det         =TWAIN string of the sensor device
OS          =User PC Operating System string
Proc        =User PC processor type and class
Speed       =User processor speed
Zip         =User postal code
```

Example:

```
<?xml version="1.0"?>
<Content>
<vendor>Digimarc</vendor>
<appl>MB</appl>
</Content>
Req=RFT
Type=1
Ser=10001
Ver=1.0
Cxt=A
Env=Q
Ctry=USA
Lang=English
HHID=1234567
Det=TWAIN string
OS=Win98
Proc=Pentium III
Speed=500
zip=74008-1234
```

Response from product handler

```
RtnCode=Success/Error number (Success = 1)
            Or
RtnCode=Success/Error number (Error <0)
MsgText=message text
```

Error reasons:

-4 Request format error – incomplete data

To provide the fastest possible system response, it is desirable that data exchanges between the originating device 12 and the remote system be as short as possible – preferably of a size that can be transported in a single internet data packet (i.e., less than about 536 bits). Such an arrangement avoids the overhead associated with data division on transmission, and data reassembly on reception.

Generally speaking, the combined elapsed time of the system service (i.e., watermark recognition by application 28c, packet delivery to router, decoding by router, handling by product handler, and return of response to application) for a single request shall average no more than 3 seconds as measured from receipt of request to 1st byte sent in response to request. Typical speeds are less than 2 seconds, with many responses being provided in less than 1 second.

Having described and illustrated the principles of our technology with reference to a preferred embodiment, it should be apparent that the technology can be modified in arrangement and detail without departing from such principles.

For example, while the detailed embodiment relies on watermark technology to identify the object 20, this is not essential. Any technology permitting machine identification of the object can be employed, including bar codes (1- or 2-D), data glyphs, and other machine-readable indicia, mag stripes, RF IDs, hash-codes based on object data, etc.

Similarly, while the detailed embodiment described the coupling between various system components as being effected by the internet, this need not be the case. Some of the linking can be effected by wireless data transmission. Other links can be conveyed through private data networks, telephone services, etc.

While the detailed router, product handler, and registration server are illustrated as comprising distinct computers, in other embodiments these functions can be consolidated in one or two computers (with functions illustrated as performed in two computers being consolidated into one, etc.), or can be distributed among more than three computers. The components can be co-located, or distributed geographically. In some

embodiments some or all of the functions performed by these components can be provided by the user's computer itself.

Likewise, the common reference to a "computer" as the device 12 shouldn't obscure the fact that the device can take many other forms. A cell phone, for example, has a processor, screen display, microphone, and wireless link that makes it well-suited for use in connection with the above-described technology. (Consider its use, for example, as the audio listening device in the arrangement disclosed in application 09/476,460.) Personal digital assistants, such as Palm Pilots and the like can be used (and accessorized for use) as device 12. Etc., etc.

While the detailed embodiment is a system of many parts, it will be recognized that novelty also resides in individual components thereof, and that such components can also be employed in other systems and devices.

To provide a comprehensive disclosure without unduly lengthening the specification, applicants incorporate by reference any patents and patent applications referenced above.

The particular combinations of elements and features in the above-detailed embodiments are exemplary only; the interchanging and substitution of these teachings with other teachings in this and the incorporated-by-reference patents/applications are also contemplated.

In view of the wide variety of embodiments to which the principles and features discussed above can be applied, it should be apparent that the detailed embodiment is illustrative only and should not be taken as limiting the scope of the invention. Rather, we claim as our invention all such modifications as may come within the scope and spirit of the following claims and equivalents thereof.

WE CLAIM:

1. A method comprising:

sensing an object identifier from a first object;

sending said first object identifier from a first device to a second device;

in response, at said second device, identifying address information corresponding to said first object identifier and sending same to the first device;

initiating a link from the first device in accordance with said address information;

at said second device, identifying additional objects related to said first object; identifying additional address information corresponding to said additional objects; and sending said additional address information to the first device;

storing said additional address information in a memory at the first device;

wherein, if an object included among said identified additional objects is sensed by the first device, the corresponding address information can be retrieved from said memory in the first device without the intervening delays of communicating with the second device.

2. A database method comprising:

generating a database record including plural data fields;

generating a file corresponding to said database record and including data from at least certain of said fields;

electronically distributing a copy of the file to each of plural recipients;

one of said recipients adding data to a copy of the file, or changing data in a copy of the file, and sending the file to the database;

updating the database record in accordance with said changed file;

generating a new file corresponding to the updated database record and including data from at least certain of said fields; and

electronically distributing a copy of the new file to each of said plural recipients.

3. A system for linking from physical or digital objects to corresponding digital resources, comprising:

registration means for receiving data relating to an object, including its identity and owner, and associating same in a database with data relating to a corresponding response;

originating device means for sensing data from an input object, processing same, and forwarding same to a routing means;

routing means for processing the processed data from the originating device means, logging information from same, and forwarding at least certain of said processed data to a product handler means; and

product handler means for providing a response to the originating device means in accordance with the information provided thereto by the routing means.

4. The system of claim 3 in which the routing means includes means for checking information in the database.

5. The system of claim 3 in which the registration means includes means for generating an encapsulating file and means for distributing said file to predetermined parties.

6. A system as described in the foregoing detailed description.

7. An apparatus including a watermark detector and a watermark-related software program, operable to transmit a packet of data to a remote system, said packet of data comprising (a) an identifier of said software program, and (b) at least a portion of a detected watermark.

8. The apparatus of claim 7, wherein said packet of data also includes address information identifying the apparatus.

9. A system comprising the apparatus of claim 7 together with said remote system, the remote system including a router and plural handlers, the router directing data from said packet to one of said handlers depending on data within said packet.

10. An apparatus including a watermark detector and a watermark-related software program, operable to transmit a packet of data to a remote system, said packet of data comprising (a) a context or environment identifier, and (b) at least a portion of a detected watermark.

11. The apparatus of claim 10, wherein said packet of data also includes address information identifying the apparatus.

12. A system comprising the apparatus of claim 10 together with said remote system, the remote system including a router and plural handlers, the router directing data from said packet to one of said handlers depending on data within said packet.

13. A networked computer system, responsive to watermark data sent from a remote client application to initiate delivery of audio or video data.

14. The networked computer system of claim 13, responsive to watermark data sent from a software program on a remote computer, to initiate delivery of audio or video data to said remote computer.

15. A networked computer system, responsive to watermark data sent from a software program on a remote computer, to initiate delivery of advertisement data to said remote computer.

16. A networked computer system, responsive to watermark data sent from a software program on a remote computer, to initiate delivery of updated software to said remote computer.

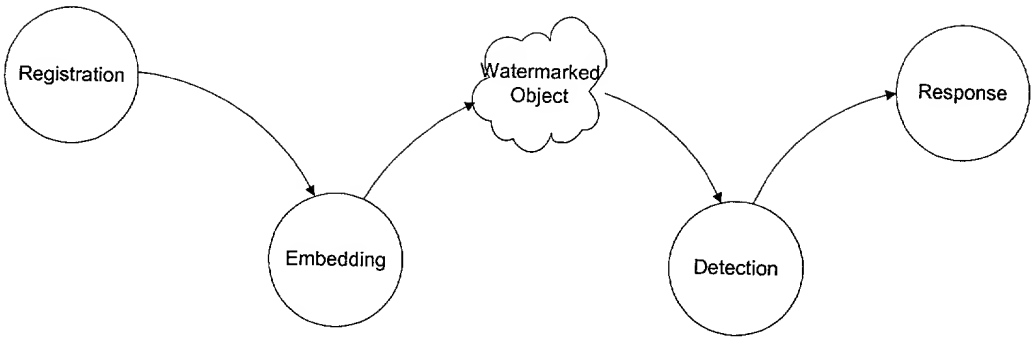
Registration
Embedding
Watermarked Object
Detection
Response

FIG. 1

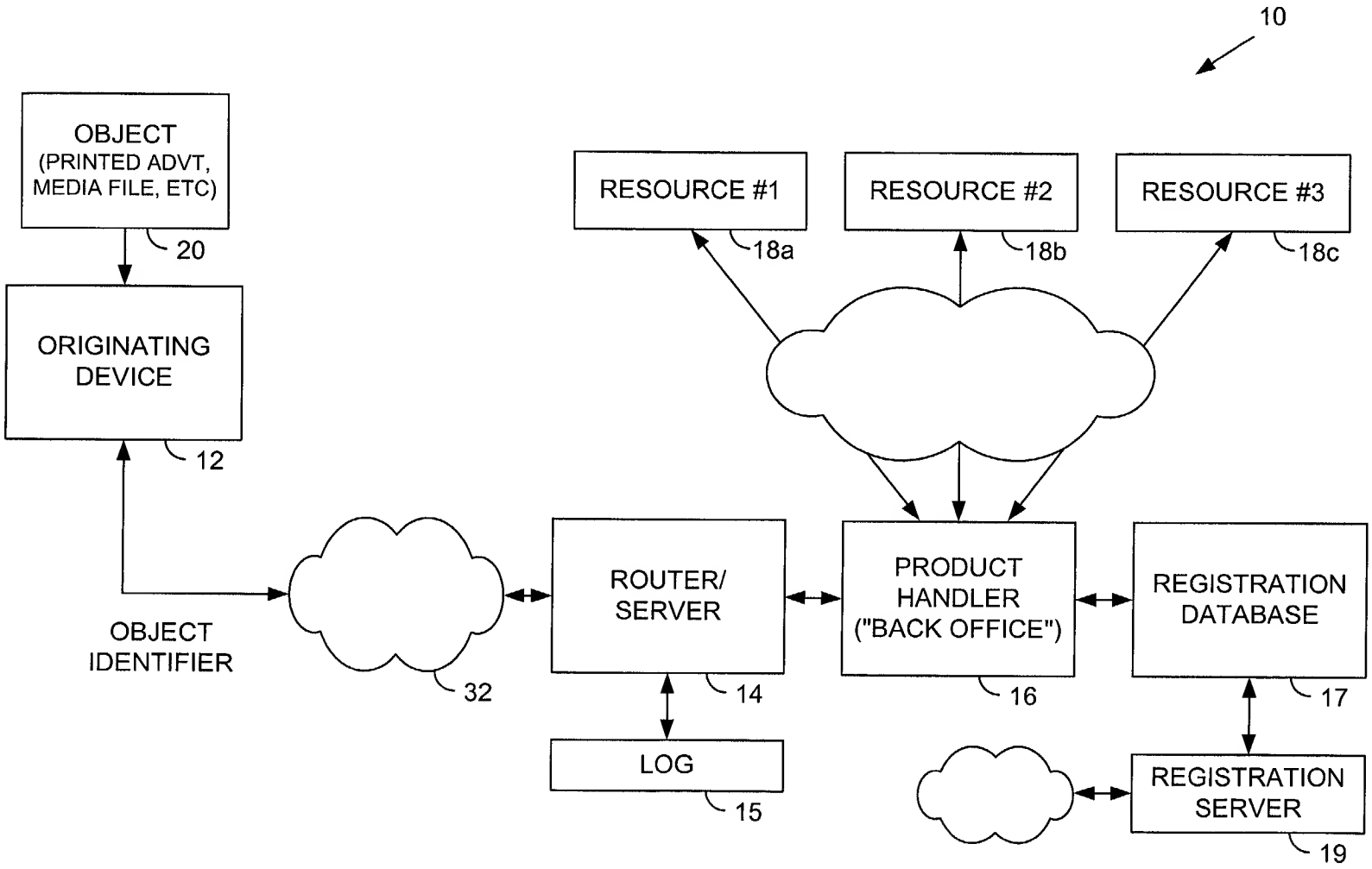
10
OBJECT
(PRINTED ADVT,
MEDIA FILE, ETC)
20
ORIGINATING
DEVICE
12
OBJECT
IDENTIFIER
32
RESOURCE #1
18a
RESOURCE #2
18b
RESOURCE #3
18c
ROUTER/
SERVER
14
LOG
15
PRODUCT
HANDLER
("BACK OFFICE")
16
REGISTRATION
DATABASE
17
REGISTRATION
SERVER
19

FIG. 2

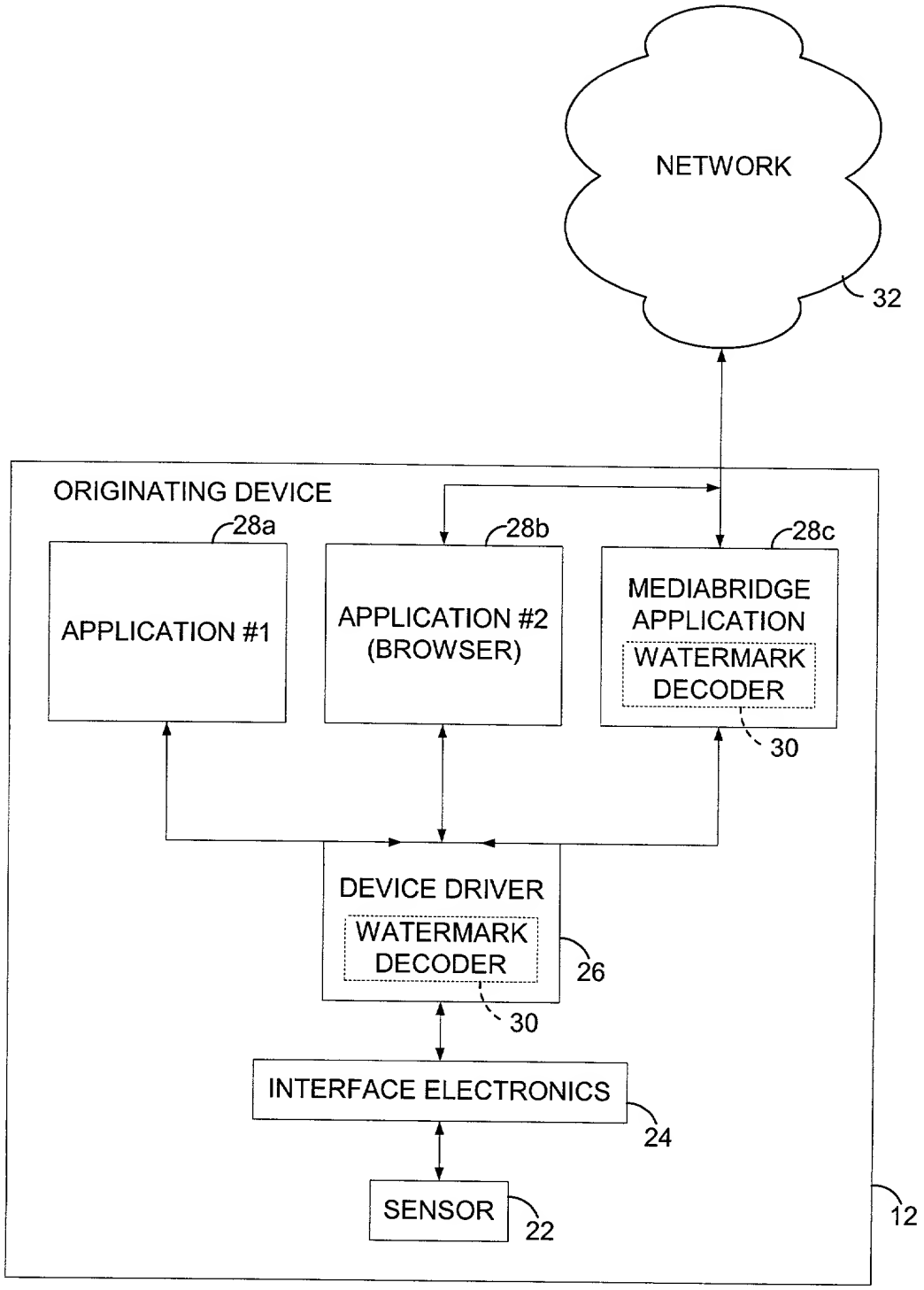
NETWORK
32
ORIGINATING DEVICE
28a
28b
28c
APPLICATION #1
APPLICATION #2
(BROWSER)
MEDIABRIDGE
APPLICATION
WATERMARK
DECODER
30
DEVICE DRIVER
WATERMARK
DECODER
26
30
INTERFACE ELECTRONICS
24
SENSOR
22
12

FIG. 3

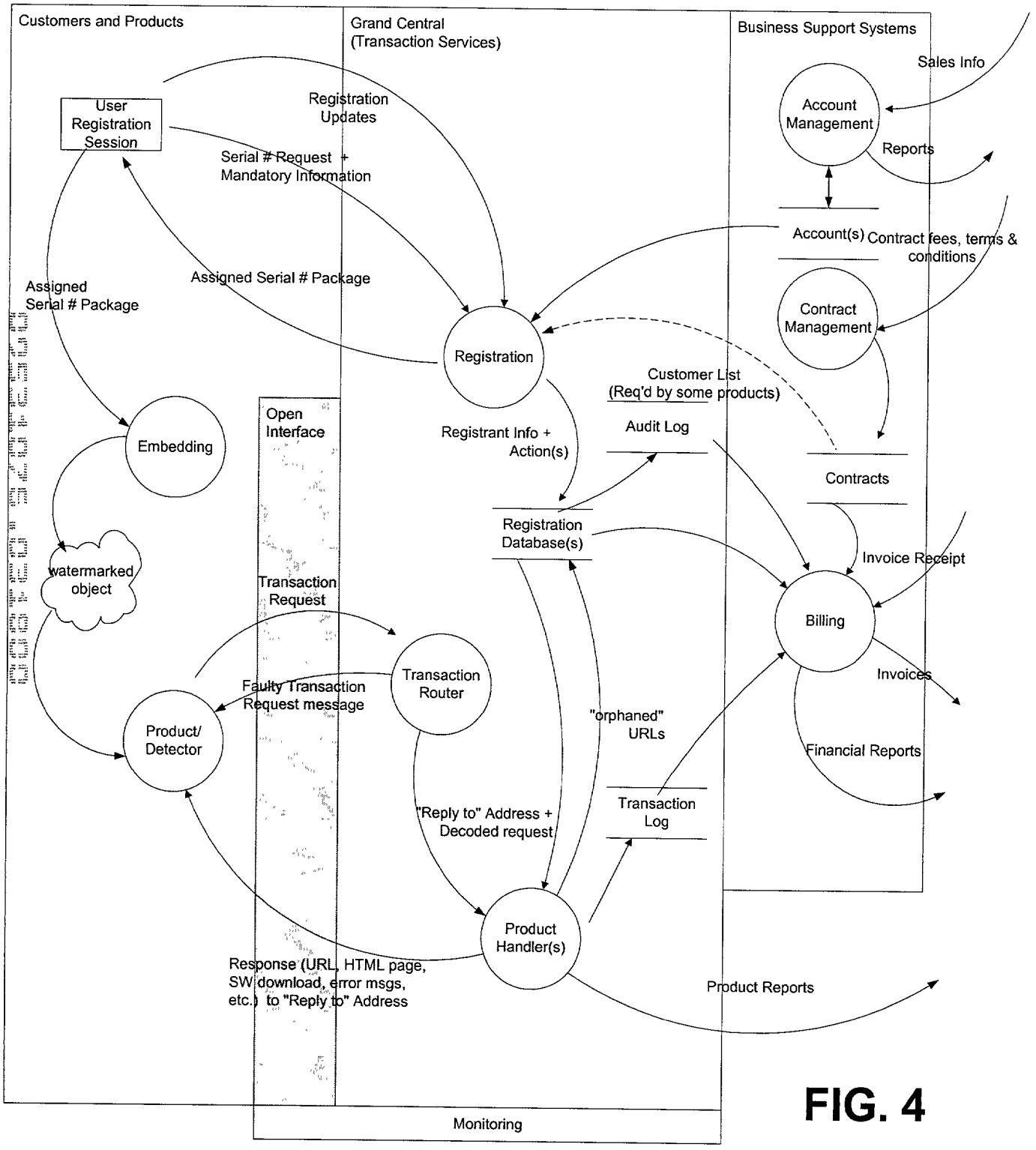
Customers and Products
Grand Central
(Transaction Services)
Business Support Systems
User Registration Session
Registration Updates
Serial # Request + Mandatory Information
Assigned Serial # Package
Assigned Serial # Package
Sales Info
Account Management
Reports
Account(s)
Contract fees, terms & conditions
Contract Management
Registration
Customer List
(Req'd by some products)
Audit Log
Registrant Info + Action(s)
Contracts
Embedding
Open Interface
Registration Database(s)
Invoice Receipt
watermarked object
Transaction Request
Billing
Transaction Router
Faulty Transaction Request message
Invoices
"orphaned" URLs
Product/ Detector
Financial Reports
Transaction Log
"Reply to" Address + Decoded request
Product Handler(s)
Response (URL, HTML page, SW download, error msgs, etc.) to "Reply to" Address
Product Reports
Monitoring

FIG. 4

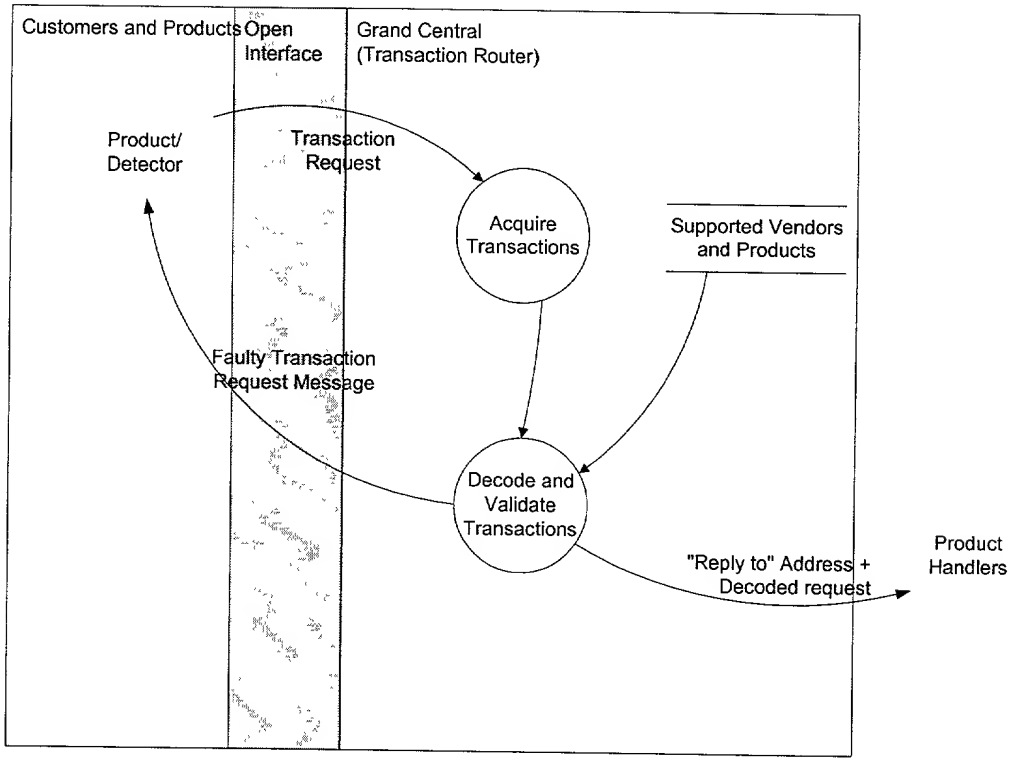
Customers and Products
Open Interface
Grand Central (Transaction Router)
Product/ Detector
Transaction Request
Acquire Transactions
Supported Vendors and Products
Faulty Transaction Request Message
Decode and Validate Transactions
"Reply to" Address + Decoded request
Product Handlers

FIG. 5

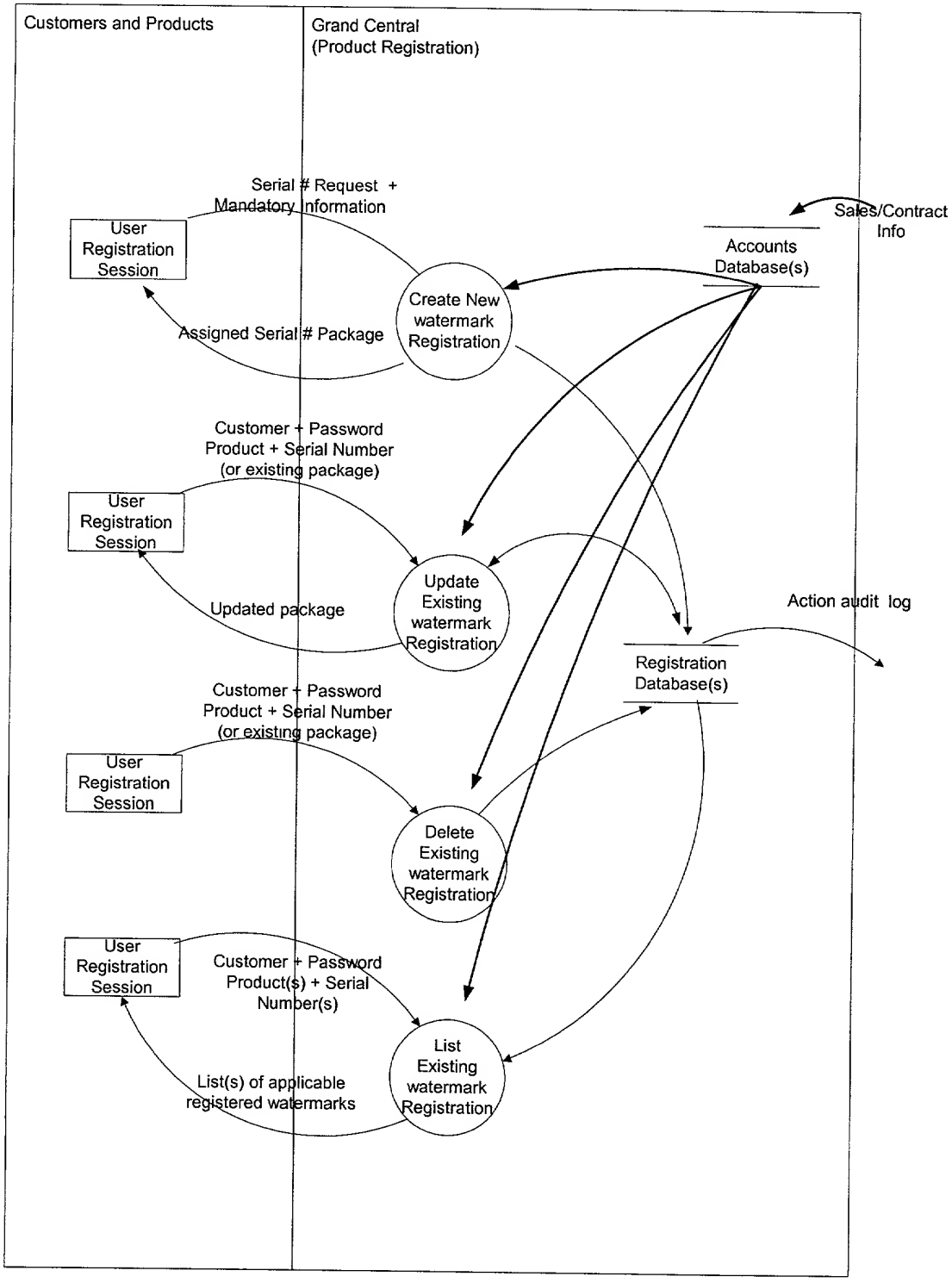
Customers and Products
Grand Central
(Product Registration)
Serial # Request +
Mandatory Information
User
Registration
Session
Create New
watermark
Registration
Assigned Serial # Package
Sales/Contract
Info
Accounts
Database(s)
Customer + Password
Product + Serial Number
(or existing package)
User
Registration
Session
Update
Existing
watermark
Registration
Updated package
Action audit log
Registration
Database(s)
Customer + Password
Product + Serial Number
(or existing package)
User
Registration
Session
Delete
Existing
watermark
Registration
User
Registration
Session
Customer + Password
Product(s) + Serial
Number(s)
List
Existing
watermark
Registration
List(s) of applicable
registered watermarks

FIG. 6

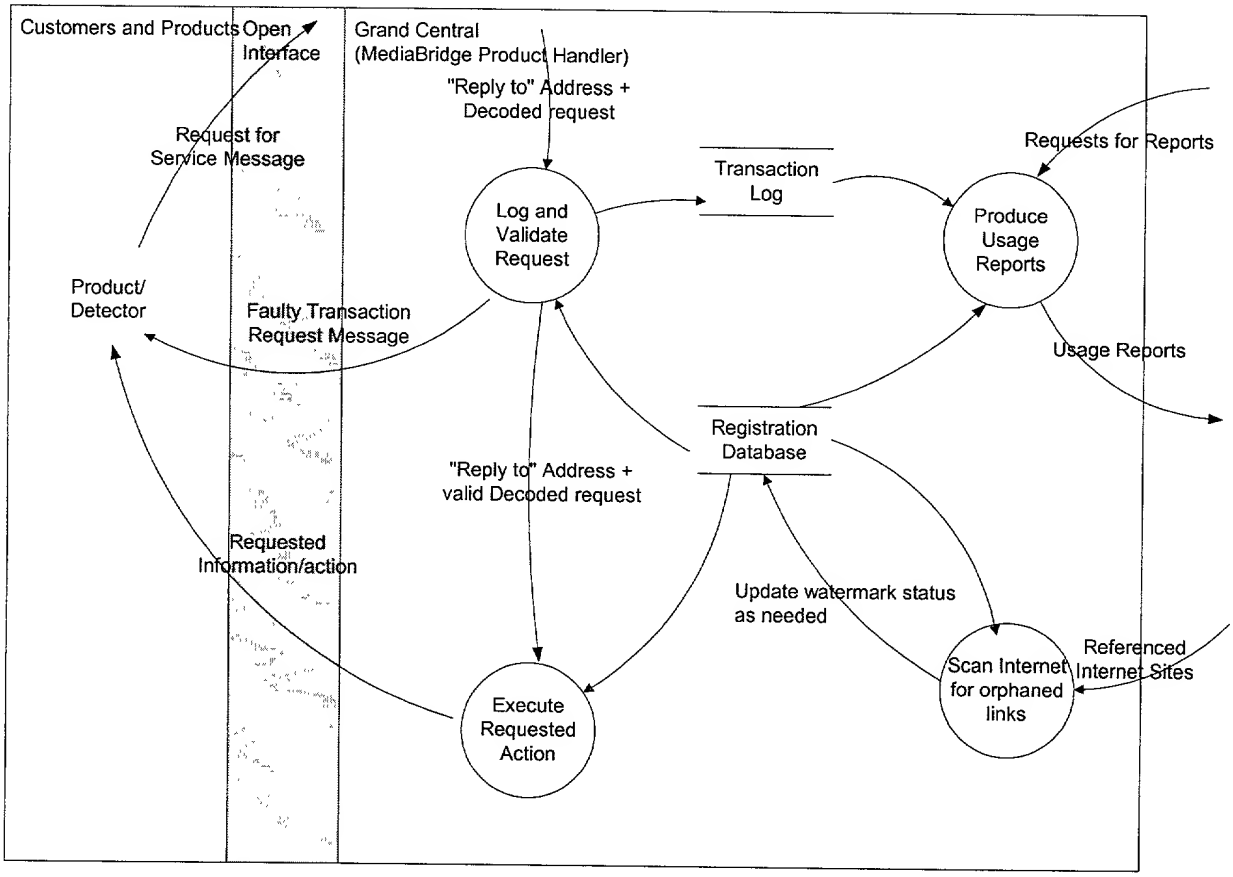
Customers and Products Open Interface
Grand Central (MediaBridge Product Handler)
"Reply to" Address + Decoded request
Request for Service Message
Transaction Log
Requests for Reports
Log and Validate Request
Produce Usage Reports
Product/ Detector
Faulty Transaction Request Message
Usage Reports
Registration Database
"Reply to" Address + valid Decoded request
Requested Information/action
Update watermark status as needed
Referenced Internet Sites
Scan Internet for orphaned links
Execute Requested Action

FIG. 7

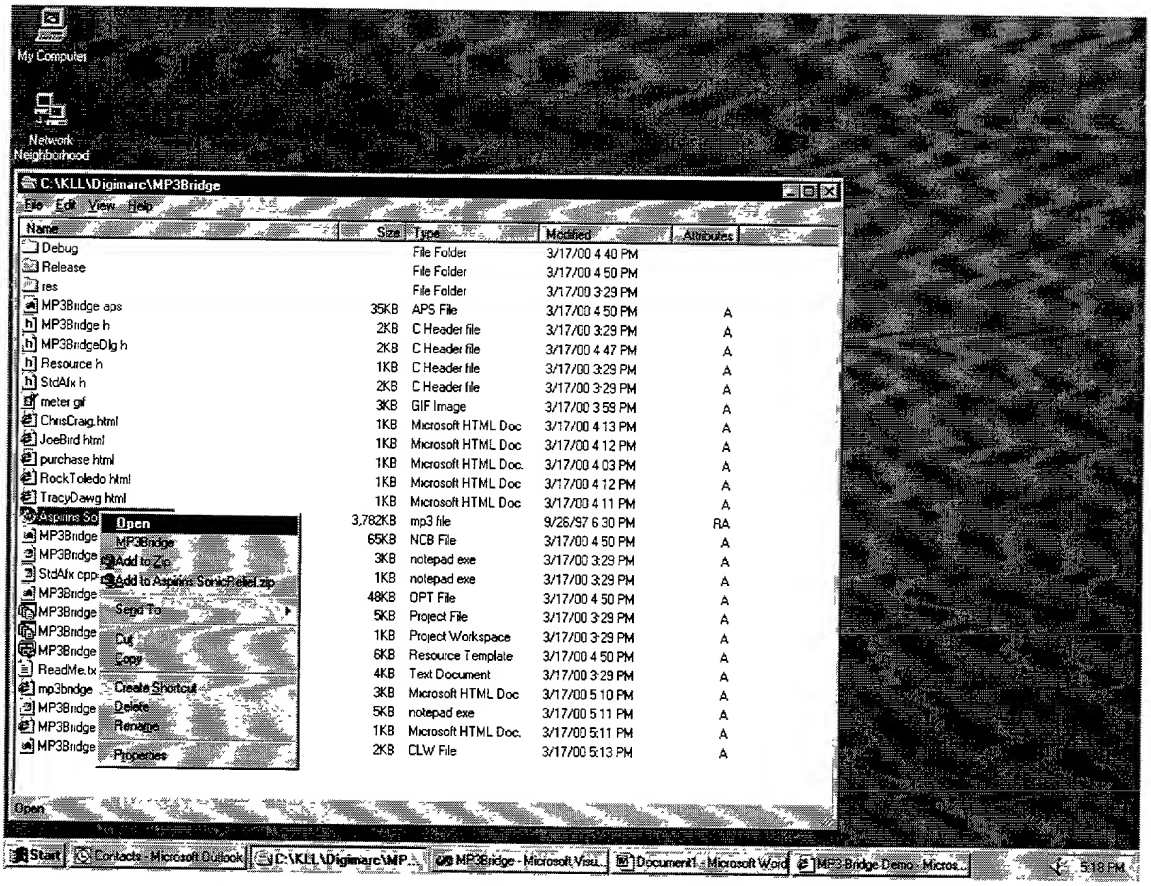
My Computer
Network Neighborhood
C:\KLL\Digimarc\MP3Bridge
File Edit View Help
Name Size Type Modified Attributes
Debug File Folder 3/17/00 4:40 PM
Release File Folder 3/17/00 4:50 PM
res File Folder 3/17/00 3:29 PM
MP3Bridge.aps 35KB APS File 3/17/00 4:50 PM A
MP3Bridge.h 2KB C Header file 3/17/00 3:29 PM A
MP3BridgeDlg.h 2KB C Header file 3/17/00 4:47 PM A
Resource.h 1KB C Header file 3/17/00 3:29 PM A
StdAfx.h 2KB C Header file 3/17/00 3:29 PM A
meter.gif 3KB GIF Image 3/17/00 3:59 PM A
ChrisCraig.html 1KB Microsoft HTML Doc 3/17/00 4:13 PM A
JoeBird.html 1KB Microsoft HTML Doc 3/17/00 4:12 PM A
purchase.html 1KB Microsoft HTML Doc. 3/17/00 4:03 PM A
RockToledo.html 1KB Microsoft HTML Doc 3/17/00 4:12 PM A
TracyDawg.html 1KB Microsoft HTML Doc 3/17/00 4:11 PM A
Aspirins.Sc 3,782KB mp3 file 9/26/97 6:30 PM RA
MP3Bridge 65KB NCB File 3/17/00 4:50 PM A
MP3Bridge 3KB notepad exe 3/17/00 3:29 PM A
StdAfx.cpp 1KB notepad exe 3/17/00 3:29 PM A
MP3Bridge 48KB OPT File 3/17/00 4:50 PM A
MP3Bridge 5KB Project File 3/17/00 3:29 PM A
MP3Bridge 1KB Project Workspace 3/17/00 3:29 PM A
MP3Bridge 6KB Resource Template 3/17/00 4:50 PM A
ReadMe.tx 4KB Text Document 3/17/00 3:29 PM A
mp3bridge 3KB Microsoft HTML Doc 3/17/00 5:10 PM A
MP3Bridge 5KB notepad exe 3/17/00 5:11 PM A
MP3Bridge 1KB Microsoft HTML Doc. 3/17/00 5:11 PM A
MP3Bridge 2KB CLW File 3/17/00 5:13 PM A
Open
MP3Bridge
Add to Zip
Add to Aspirins SonicRebel.zip
Send To
Cut
Copy
Create Shortcut
Delete
Rename
Properties
Open
Start
Contacts - Microsoft Outlook
C:\KLL\Digimarc\MP...
MP3Bridge - Microsoft Visu...
Document1 - Microsoft Word
MP3 Bridge Demo - Micros...
5:18 PM

FIG. 8

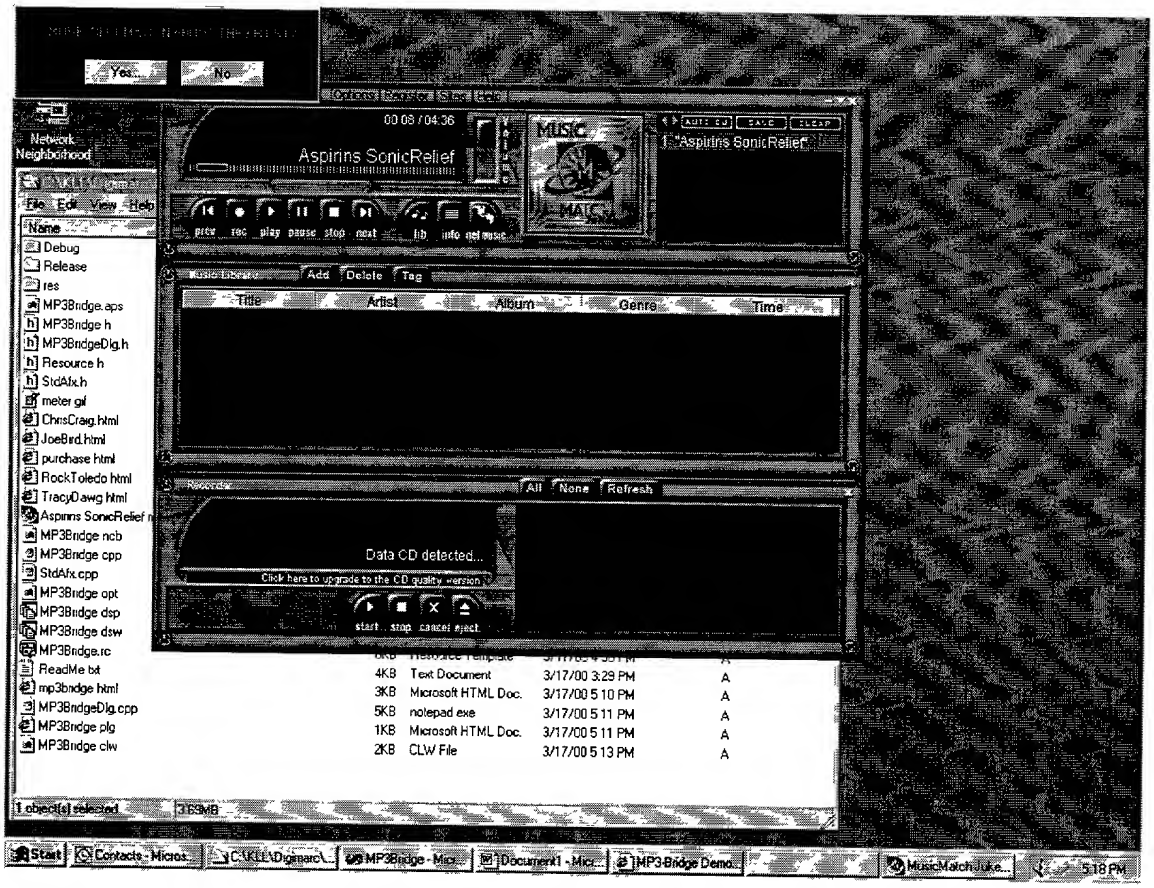
Yes...
No
Network Neighborhood
00:08 / 04:36
Aspirins SonicRelief
1. "Aspirins SonicRelief"
Add
Delete
Tag
Title
Artist
Album
Genre
Time
All
None
Refresh
Data CD detected...
Click here to upgrade to the CD quality version
start
stop
cancel
eject
Debug
Release
res
MP3Bridge.aps
MP3Bridge.h
MP3BridgeDlg.h
Resource.h
StdAfx.h
meter.gif
ChrisCraig.html
JoeBird.html
purchase.html
RockToledo.html
TracyDawg.html
MP3Bridge.ncb
MP3Bridge.cpp
StdAfx.cpp
MP3Bridge.opt
MP3Bridge.dsp
MP3Bridge.dsw
MP3Bridge.rc
ReadMe.txt
mp3bridge.html
MP3BridgeDlg.cpp
MP3Bridge.plg
MP3Bridge.clw
4KB Text Document 3/17/00 3:29 PM A
3KB Microsoft HTML Doc. 3/17/00 5:10 PM A
5KB notepad.exe 3/17/00 5:11 PM A
1KB Microsoft HTML Doc. 3/17/00 5:11 PM A
2KB CLW File 3/17/00 5:13 PM A
1 object(s) selected
Start
Contacts - Micros...
MP3Bridge - Micr...
Document1 - Mic...
MP3Bridge Demo...
MusicMatch Juke...
5:18 PM

FIG. 9

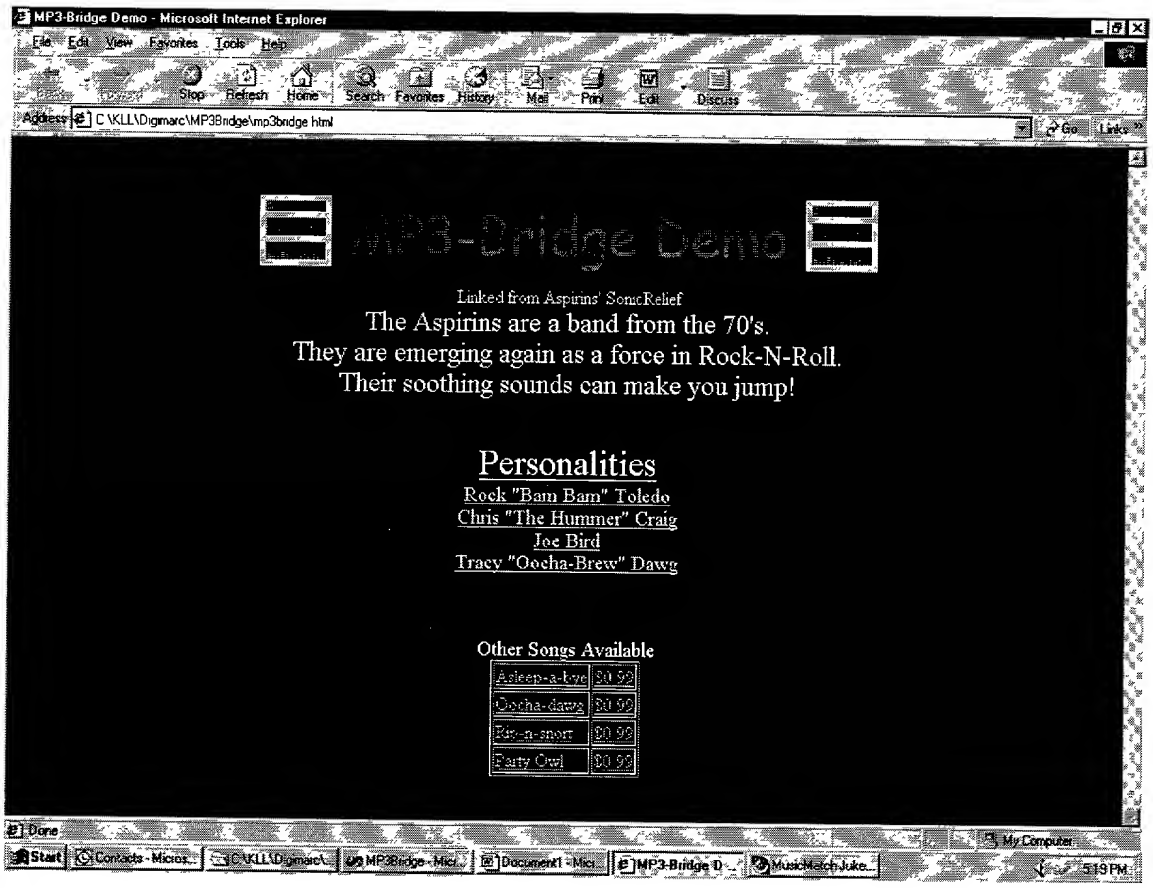
MP3-Bridge Demo - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Stop Refresh Home Search Favorites History Mail Print Edit Discuss
Address C:\KLL\Digimarc\MP3Bridge\mp3bridge.html
Go Links
MP3-Bridge Demo
Linked from Aspirins' SonicRelief
The Aspirins are a band from the 70's.
They are emerging again as a force in Rock-N-Roll.
Their soothing sounds can make you jump!
Personalities
Rock "Bam Bam" Toledo
Chris "The Hummer" Craig
Joe Bird
Tracy "Oocha-Brew" Dawg
Other Songs Available
Asleep-a-bye $0.99
Oocha-dawg $0.99
Rit-a-snort $0.99
Party Owl $0.99
Done
My Computer
Start
Contacts - Micros...
C:\KLL\Digimarc\
MP3Bridge - Micr...
Document1 - Micr...
MP3-Bridge D...
MusicMatch Juke...
5:19 PM

FIG. 10